WHITE PAPER

Porting Applications from Compaq Visual Fortran to Intel® Visual Fortran Compilers

Intel® software development products

Steven Lionel
Intel Compiler Support
Intel Corporation

June 16, 2005

Contents

Executive Summary

The Intel Visual Fortran Compiler for Windows*, available in a Standard Edition and a Professional Edition, is the next-generation successor to Compaq Visual Fortran, combining the technologies of the Intel and Compaq compilers. This paper introduces the necessary background for developers who are migrating to the Intel Visual Fortran Compilers from Compaq Visual Fortran. It discusses changes in behavior, and it highlights source and build changes that may be needed as part of that migration. Unless otherwise noted, the information provided here is relevant to both the Standard and Professional Editions of the Intel Visual Fortran Compilers.

Introduction

The Intel® Visual Fortran Compiler 9.0 for Windows* integrates with Microsoft developer tools to provide very high levels of optimization for the full spectrum of Intel® processor technologies. In addition to performance and industry-compatibility gains, this version provides full support for Intel multi-core processors and IA-32 processors supporting Intel® EM64T, as well as support for previous generations of hardware platforms. Building on the advances in version 8.x, Intel Fortran Compiler 9.0 for Linux* and Intel Visual Fortran Compiler 9.0 for Windows represent a joining of Fortran technologies from Intel and Compaq (now part of Hewlett-Packard). Intel code generation, optimization, and parallel processing technologies combine with the features, extensions, and language processing of Compaq Visual Fortran to create Fortran compilers that offer a robust feature set, along with unmatched runtime performance.

In most cases, you can rebuild existing applications with the new compilers without source changes, but some applications may need minor coding changes, and build methods may need minor adjustments. This paper describes the key differences you are likely to encounter. For additional details, refer to the appropriate compiler release notes and the compiler documentation.

This paper applies to those coming to the Intel Visual Fortran Compiler 9.0 from Compaq Visual Fortran (CVF). If you are porting from Intel Fortran Compiler 7.1, please refer to the separate document, *Migrating Applications from Intel® Fortran Compiler 7.1 to Intel Fortran Compiler 9.0.*

Product Compatibility

Intel Visual Fortran Compiler can coexist on a system with CVF, and you can continue using the older product if you wish. However, Intel Visual Fortran Compiler does not integrate with Microsoft Visual Studio* 6, whereas CVF does not integrate with Microsoft Visual Studio .NET. The Intel and Compaq products install into separate folder trees and use separate registry variables.

All Fortran sources must be recompiled with Intel Visual Fortran Compiler; you cannot use CVF-compiled objects, modules, or static libraries with Intel Visual Fortran Compiler. You can, however, use CVF-built dynamic link libraries (DLLs) with applications compiled with the Intel Visual Fortran Compiler, as long as you do not try to share input/output units across the two environments. Note also that third-party libraries built for use with CVF may not work with Intel Visual Fortran Compiler. Contact the library supplier for more information.

Microsoft Visual C++* .NET Required

In order to install and use the Intel Visual Fortran Compiler, you must have one of the following Microsoft development products already installed:

- Microsoft Visual C++* .NET 2002 or 2003, Standard Edition or higher
- Microsoft Visual Studio* .NET 2002 or 2003 (Visual C++ component required)

Installation of the Microsoft development products provides necessary libraries and tools used by the Intel Visual Fortran Compiler, as well as the visual development environment. Please make sure that you have an appropriate Microsoft development product installed before attempting installation of the Intel Visual Fortran Compiler.

Common Features

Intel Visual Fortran Compiler supports all of the CVF language syntax, including extensions from Digital Equipment Corporation (DEC) Fortran and Microsoft Fortran PowerStation* 4. All CVF library routines are supported, including those from the QuickWin and Portability libraries, as are all of the system-interface modules. In most cases, a simple rebuild of the application with the Intel compiler is all that is needed. Source changes that you may need to make are described below.

Features Not Supported

Intel Visual Fortran Compiler supports all of the language syntax supported by CVF. However, some of the CVF product features are not supported by Intel Visual Fortran Compiler. These include the following:

- Compaq Extended Math Library. The Intel® Math Kernel Library or third-party libraries such as IMSL* and NAG* may be suitable alternatives. (IMSL is provided in Intel Visual Fortran Compiler, Professional Edition; other products must be obtained separately.)
- Source Browser cross-reference tool
- COM Server Wizard from CVF Professional Edition

If you have existing applications that were created by the CVF COM Server Wizard, you may be able to rebuild them with the Intel Visual Fortran Compiler. If you need to make changes to the interfaces, you can do so in CVF if you have left it installed.

Migrating CVF Projects

Intel Visual Fortran Compiler includes project-conversion wizards to make it easy to migrate from CVF. Conversion is a two-step process:

1. Open the CVF workspace in Visual C++* .NET by right-clicking on the workspace's **.DSW** file and selecting **Open With... Microsoft Visual Studio .NET**. You will see a message similar to that in Figure 1. Click **Yes To All** to convert each project to a Visual C++ .NET project in a "solution" (similar to a workspace).

2. In the right pane, you will see the **Solution Explorer** tab with the project(s) present, as seen in Figure 2. but the conversion to Fortran project(s) will not be complete yet. For each project, right-click the project name and select **Extract Compaq Visual Fortran Project Items**.

The project will now be converted as seen in Figure 3.

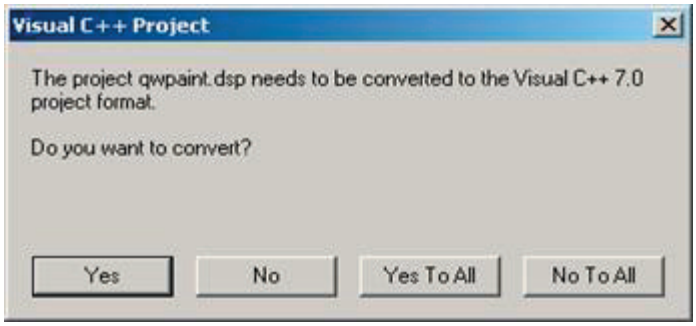

Figure 1. Initial Project Conversion Dialog

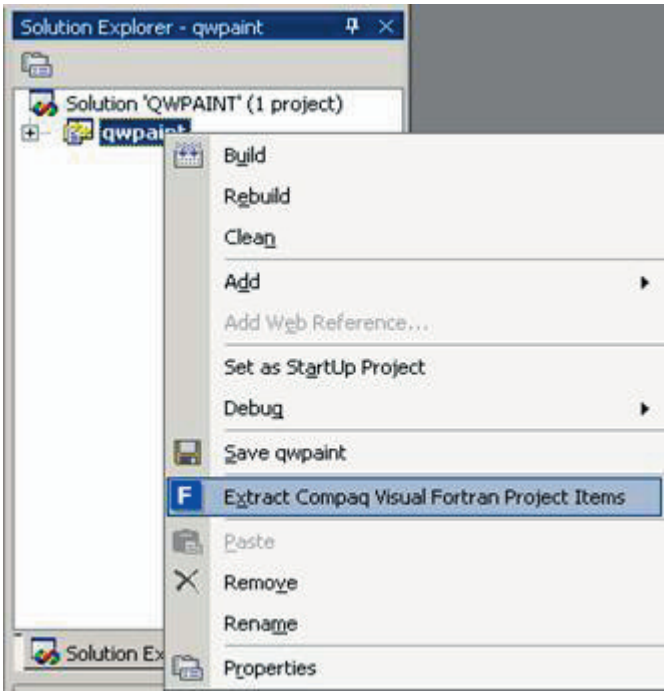

Figure 2. Extract Fortran Project Items Dialog

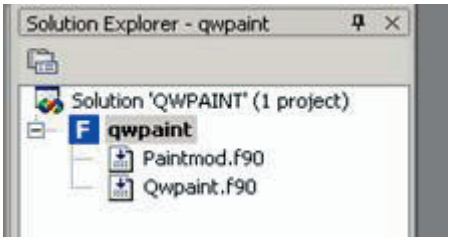

Figure 3. Converted Project

If the CVF project contains both Fortran and C sources, it must be converted into two single-language projects under a solution — one builds a static library and the other links to that library — because Microsoft Visual Studio .NET does not allow multiple languages in a single project. The project conversion wizard asks you which language has the main (linkable) project, Fortran or C, and makes the appropriate adjustments. Figure 4 is an example of converting a mixed-language project.

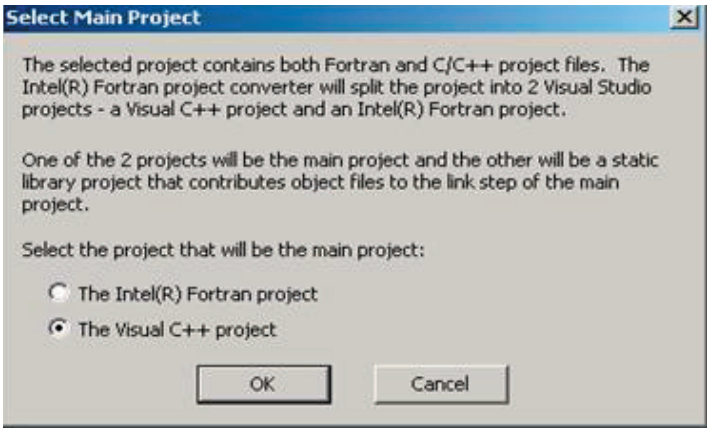

Figure 4. Mixed Language Project Dialog Box

After conversion, the mixed-language project looks like Figure 5.

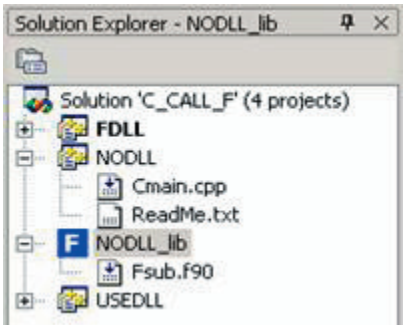

Figure 5. Converted Mixed Language Project

In the prior example, the CVF NODLL project was split into a C++ executable project named NODLL and an Intel Fortran static library project NODLL.lib. The conversion wizard automatically makes NODLL_lib a dependant of NODLL, so that the library is built first and then is linked into the C code.

If you convert a CVF project, the conversion wizard will change project settings from the default to enhance compatibility with CVF. An important change is the default calling convention: if it was "Default" in CVF, the conversion wizard changes it to "CVF." (The following section provides more information on calling conventions.) Unless you had a mixed-language application that depended on CVF-specific calling conventions, you can set the default calling convention back to the default, in most cases.

Getting Used to the Microsoft Visual C++ *.NET IDE

The Microsoft Visual C++ .NET Integrated Development Environment (IDE) is so different from the one shared by CVF that some users may find it difficult to perform common tasks. This paper covers some of the major changes, but it is not comprehensive. For more information on using the IDE, see the MSDN documentation that accompanies Visual C++ .NET or Microsoft Visual Studio .NET.

Projects and Solutions

In CVF (and Visual C++ 6.0), "projects" were placed in "workspaces." A CVF workspace was little more than a container for one or more projects and was not involved in the build process. A project built something (an EXE, LIB or DLL file in most cases), and it could contain both Fortran and C code. One project was always designated as "active."

In Visual C++ .NET, projects are substantially the same, but a project can be associated with only one language. For example, if you add C files to a Fortran project, the C files will be ignored. A solution holds multiple projects, but it is different from a workspace because you can build a solution, which builds all of the contained projects in a specified, user-configurable order.

When you have a mixed Fortran and C application, you must put the Fortran code into a Fortran project and the C code into a C (or C++) project. The projects get built separately and then, if appropriate, they are linked together. If the old project was a static library, two static library projects are created, with the objects going into a combined .LIB file. In this case, it does not matter which project you select as being the "main" project.

Changing Settings

Changing settings in the Visual C++ .NET IDE is different as well. Instead of a tabbed dialog box for Settings, there is a tree-view set of "Property Pages." Figure 6 shows a set of property pages for an example project. In this example, the Fortran General Property Page is displayed. Current property values that are the defaults are shown bolded, and a brief description of the highlighted property is displayed at the bottom of the pane.

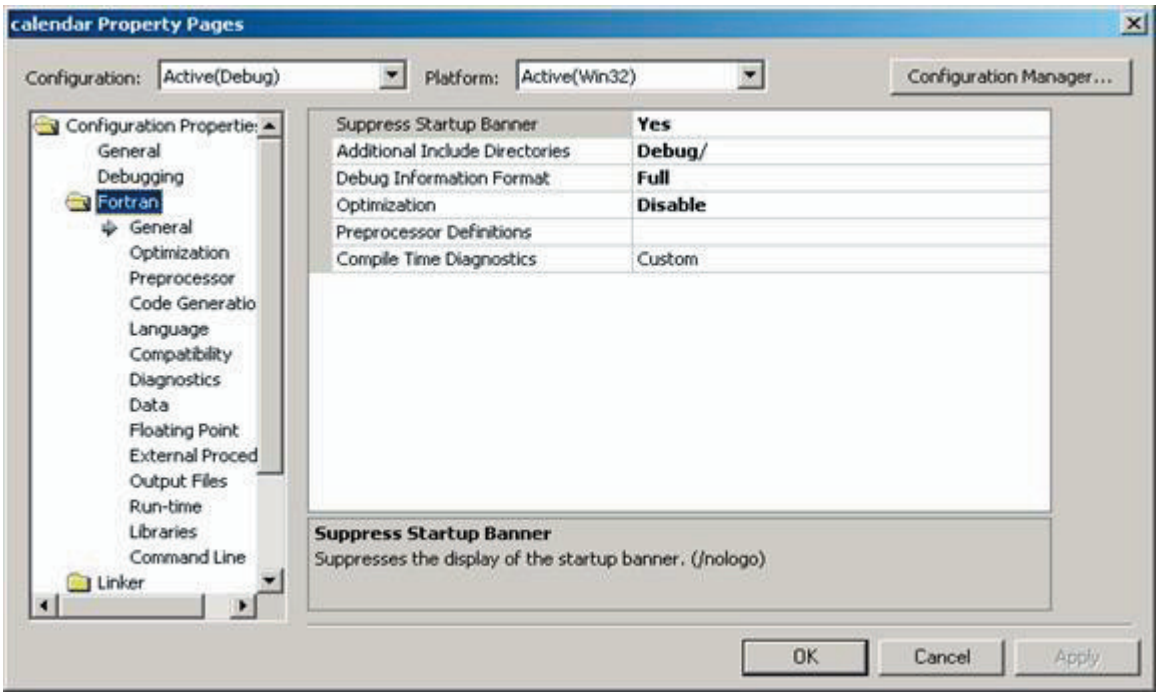

Figure 6. Property Pages

1. To change a property value that has specific options, click on the value. An arrow icon displays to the right of the value, as seen in Figure 7.
2. Click on the arrow to display the options and select the desired option. Properties that offer a list of items, such as Additional Include Directories in the example above, display a list icon (three dots).
3. Click the icon to open a separate dialog box, where you can enter the values. If you have only one value, type it directly on the Property page. You can directly type properties that are a single text string. In some cases, an arrow icon is available to allow you to select "Inherit from project defaults."

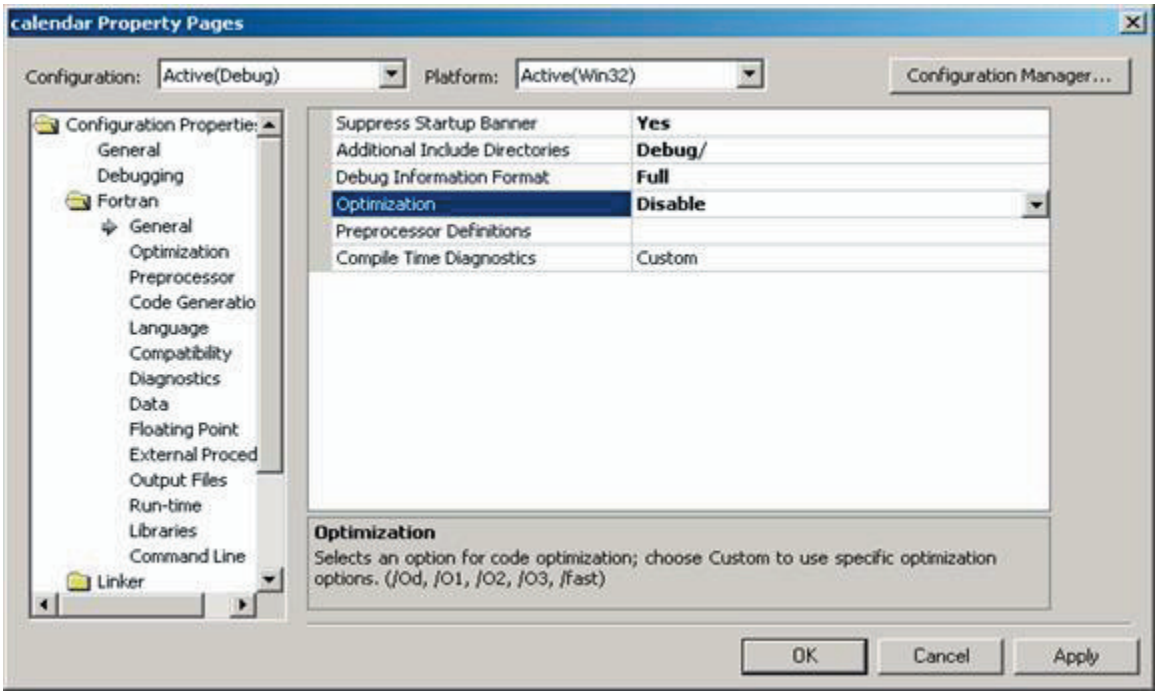

Figure 7. Changing a Property

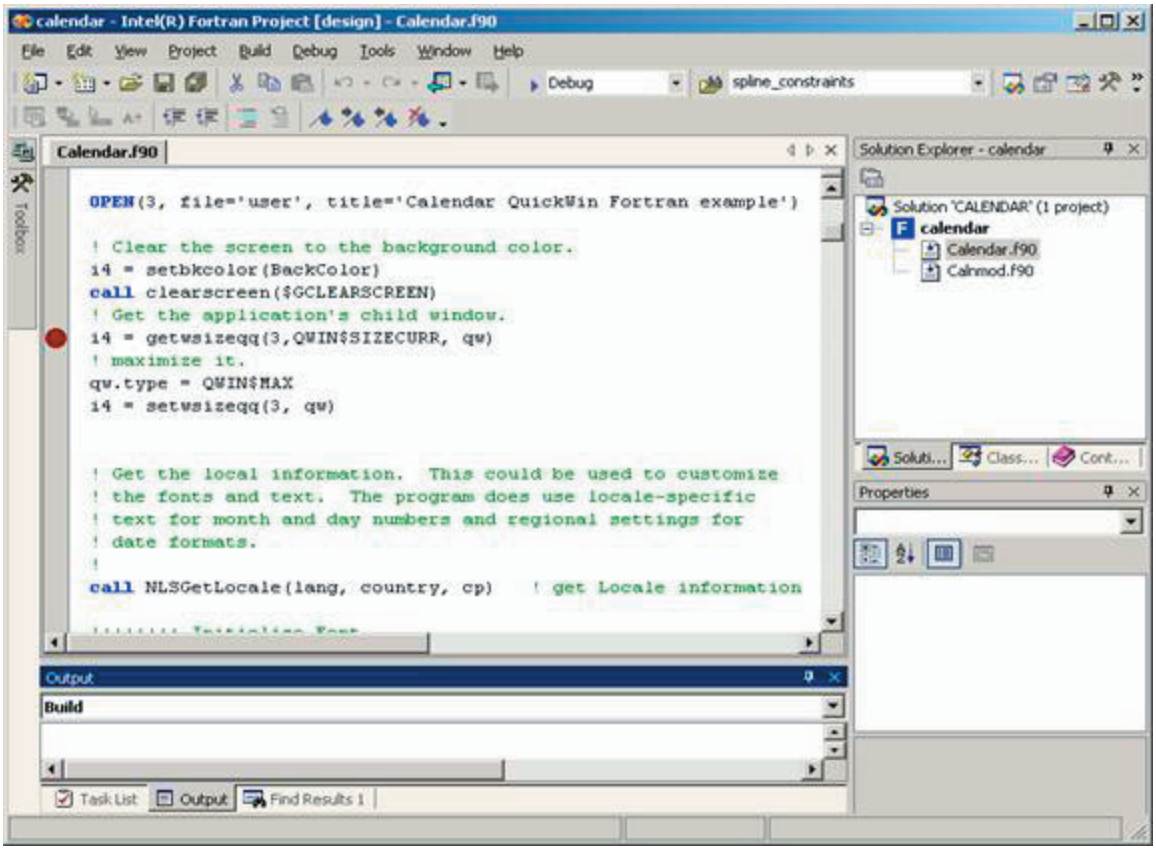

Figure 8. Intel® Debugger

Debugging

Debugging with the Intel Visual Fortran Compiler is similar to using CVF, but some of the controls are in different places as seen in Figure 8.

- To set a breakpoint, click in the left margin next to a statement; the red Stop icon displays.
- To execute under the debugger, click the blue triangle "Play" icon next to the configuration name.

Source Changes

While Intel has taken great care to avoid the need for source changes to permit "rebuild and go," a number of implementation differences between the Intel and Compaq compilers necessitate making changes for selected applications.

While the part of the compiler that handles Fortran syntax and semantics, often referred to as the "front end," is derived from CVF, improvements have been made in detecting incorrect usage, and you may find that the new compiler issues diagnostic messages for certain usages where the older compiler did not. For example, the compiler now gives an error for a source that makes a call to a non-pure intrinsic such as RANDOM_NUMBER from inside a pure procedure. In such cases, you will need to correct the coding errors. If you believe a diagnostic is inappropriate, please contact Intel® Premier Support.

Default Calling Conventions Have Changed

In CVF, the default calling mechanism was STDCALL, and routine names were "decorated" by adding @n to the end, where "n" was the number of bytes of argument list. Intel Visual Fortran Compiler adopts the more common C calling mechanism used by versions 7.1 and earlier of the Intel Fortran Compiler. Routine names are still converted to uppercase by default, and a leading underscore is added, but there is no @n suffix.

Another change is the manner in which CHARACTER argument lengths are passed. In CVF, these were passed immediately following the address of the CHARACTER item, but in Intel Visual Fortran Compiler, all the lengths are passed at the end of the argument list. In other words, the default has changed from `/iface:mixed_str_len_arg` to `/iface:nomixed_str_len_arg`.

If you have a Fortran-only application, this change may not be important to you. If, on the other hand, you have a mixed-language application, you need to be aware of the impact of the change in compiler default. You can tell the compiler to use the CVF default on the External Procedures property page, or with the `/iface:cvf` command-line switch.

GETARG, IARGC and NARGS Are Now Intrinsic

The command-line inquiry routines `GETARG`, `IARGC` and `NARGS` are now recognized as intrinsic procedures by the compiler. If your application source declares any of these names as `EXTERNAL` or provides an explicit procedure interface for them, you must remove those declarations to prevent link-time errors.

New Module Names for System and Library Declarations

Compaq Visual Fortran provided modules with definitions of Win32* API routines and symbols, as well as modules for Fortran library routines. These modules had names such as DFWIN, DFLIB, etc. Intel Visual Fortran Compiler provides compatible modules with the same names as seen in Table 1, but these are wrappers around modules with new names. You do not need to change your sources, but you should begin using the

Table 1: Old and New Names of System and Library Modules

Old Name	New Name	Description
DFAUTO	IFAUTO	Automation
DFCOM	IFCOM	COM, OLE
DFCOMITY	IFCOMTY	Obsolete; use IFWINTY
DFLIB	IFCORE, IFPORT, IFQWIN	General library, Portability library, QuickWin
DFLOGM	IFLOGM	Dialog
DFLOGMT	IFLOGMT	Dialog types and constants
DFMT	IFMT	Multithread routines
DFNLS	IFNLS	National Language Support
DFOPNGL	IFOPNGL	OpenGL routines
DFOPNGLT	IFOPNGLT	OpenGL types and constants
DFPORT	IFPORT	Portability routines
DFWBASE	IFWBASE	Deprecated WIN16 routines
DFWIN	IFWIN (or individual modules such as KERNEL32, USER32)	WIN32 API routines
DFWINA	IFWINA	Renamed WIN32 routines which conflict with QuickWin routines
DFWINTY	IFWINTY	WIN32 API types and constants

new names in new development. In DFLIB, symbols have been relocated into one of three new modules: IFCORE, IFPORT and IFQWIN. You may find it useful to select the specific module containing the symbols you are interested in.

The individual WIN32 API modules such as KERNEL32 have the same names in Intel Visual Fortran Compiler as in Compaq Visual Fortran.

Build Changes

In most cases, you will not need to make changes in your build procedures. However, to provide for compatibility with future versions of Intel Fortran compilers, some changes are recommended. This section describes differences that affect building applications.

Compile Command is Now `ifort`

Under CVF, four command names were provided for invoking the compiler: `df`, `f90`, `f77` and fl32. df and `f90` were equivalent, `f77` added options for compatibility with Compaq Fortran 77, and `fl32` added options for compatibility with Microsoft Fortran PowerStation. In Intel Visual Fortran Compiler, `ifort` is the preferred command name to invoke the compiler. `df` is also accepted, but gives a warning that can be suppressed with /quiet. `f77` and `fl32` are not provided.

If you leave CVF installed on your system, it coexists with the Intel Visual Fortran Compiler. When using the command line, be sure to use the appropriate shortcut in the Start menu to start your command session. For CVF, use “Fortran Command Prompt.” For the Intel Visual Fortran Compiler, it is “Fortran Build Environment for IA-32 Applications.” Each of these will establish the proper environment for the selected compiler. The IDEs are separate and do not interfere with each other.

Using the IMSL* Fortran Libraries from Visual Numerics

Intel Visual Fortran Compiler, Professional Edition, includes the IMSL Fortran Library 5.0 from Visual Numerics, Inc. In addition to many new and updated routines, the new version adds the following features not included in Compaq Visual Fortran Professional Edition:

- Multithreaded libraries for improved performance on multi-core and multiprocessor systems and Intel processors with Hyper-Threading Technology.
- DLL form of the libraries.
- Support for the Intel Itanium® processor and processors supporting Intel EM64T.
- New, unified interface modules making use of Fortran 95 generic interfaces and optional arguments.

In Intel Visual Fortran Compiler, Professional Edition, the IMSL libraries are installed separately, after the compiler is installed.

An existing application that uses the IMSL libraries will need few or no changes — unless you want to take advantage of the new features. You will need to make a change in how the application is built to reference the proper libraries. Below is an overview of this change. See the Intel Visual Fortran Compiler User’s Guide for additional details.

Locating and Referencing the IMSL Libraries

In Compaq Visual Fortran, there was just one set of IMSL libraries: static and non-threaded. In Intel Visual Fortran Compiler, Professional Edition, there are four sets of libraries with different names. Rather than explicitly list the libraries, as was commonly done with CVF, you should instead specify the path to the folder containing the libraries and use one of the supplied INCLUDE files to create a reference to the specific libraries required.

In the Visual C++ .NET IDE, add the path to the IMSL libraries by selecting the menu options **Tools** > **Options** > **Intel Fortran** and inserting in the **Project Directories: Libraries** list the path to the IMSL libraries (typically `C:\Program Files\VNI\CTT6.0\LIB\IA32`). If you are using the command line, after starting the command prompt session, run "`C:\Program Files\VNI\CTT6.0\ctt\bin\CTTSETUP.BAT`" to define the LIB environment variable appropriately.

Do the same for the IMSL INCLUDE folder, adding it to the list under **Project Directories: Includes**. If you use the command line, the INCLUDE environment variable is set automatically by CTTSETUP.BAT or during installation.

The remaining step causes the appropriate libraries to be referenced by adding one of the following lines to any one source file in your application:

- `INCLUDE 'link_f90_static.h'`
  –uses the single-processor, static library form of the libraries.
- `INCLUDE 'link_f90_dll.h'`
  –uses the single-processor, dynamic link library form of the libraries.
- `INCLUDE 'link_f90_static_smp.h'`
  –uses the multiprocessor, static library form of the libraries (the DLL form of the Intel® Math Kernel Library will always be used).
- `INCLUDE 'link_f90_dll_smp.h'`
  –uses the multiprocessor, dynamic link library form of the libraries.

For additional details, refer to the User's Guide section on using IMSL.

Product Features

Intel Visual Fortran Compiler provides a rich feature set that delivers winning performance for both legacy and cutting-edge technologies:

- Full support of Intel multi-core processors and Intel processors supporting Intel EM64T, along with previous Intel processors and architectures.
- Quadruple-precision floating point REAL(16) and COMPLEX(32).
- Automatic parallelization.
- OpenMP* support.
- Advanced optimization for new Intel processors.
- Code Coverage and Test Prioritization tools.
- Command Line Debugger (Intel® Debugger).

For more information on these and other features, see the compiler Release Notes.

Getting Help

For "how to" questions, visit the Intel Fortran user forums at http://softwareforums.intel.com/ to ask questions of other knowledgeable users and search for previous discussions that may address your problem.

If you believe you have found a bug in the product, contact Intel Premier Support. You must first register for support; you are prompted to register when you install the compiler, and you may also register at the Registration Center at http://www.intel.com/software/products/registrationcenter/index.htm. Once registered, log in at https://premier.intel.com/ and submit your support request.

For additional information about Intel® compilers and other Intel® Software Development Tools, see the Intel® Software Development Products Web site at http://www.intel.com/software/products/ and the Intel® Software Tools Developer Center at http://www.intel.com/cd/ids/developer/asmo-na/eng/technologies/tools/index.htm.

Intel Corporation
2200 Mission College Blvd.
Santa Clara, CA 95052-8119
USA

For product and purchase information visit:

www.intel.com/software/products